An effective fractal image compression algorithm based on plane fitting*

Wang Xing-Yuan(王兴元)†, Guo Xing(国 兴), and Zhang Dan-Dan(张丹丹)

Faculty of Electronic Information & Electrical Engineering, Dalian University of Technology, Dalian 116024, *China*

(Received 8 February 2012; revised manuscript received 16 March 2012)

A new method using plane fitting to decide whether a domain block is similar enough to a given range block is proposed in this paper. First, three coefficients are computed for describing each range and domain block. Then, the best-matched one for every range block is obtained by analysing the relation between their coefficients. Experimental results show that the proposed method can shorten encoding time markedly, while the retrieved image quality is still acceptable. In the decoding step, a kind of simple line fitting on block boundaries is used to reduce blocking effects. At the same time, the proposed method can also achieve high compression ratio.

Keywords: fractal image compression, plane fitting, peak signal to noise ratio, gray image

PACS: 05.45.Df, 07.05.Pj, 42.30.Wb **DOI:** 10.1088/1674-1056/21/9/090507

1. Introduction

Fractal image compression was proposed by Barnsley[1] in 1988 and implemented by Jacquin[2] in 1992. Partitioned iterated function system (PIFS) and collage theorem form its basis. By utilizing the self-similarity characteristic in a natural image, fractal image compression can achieve a high compression ratio and good retrieved image quality. However, the calculation of similarity between a range block and a domain block in the encoding step is very complex and time-consuming. Many encoding techniques have been presented by the researchers to overcome this problem.[3−14] In 2009, Ameer *et al.*[15] developed a simple scheme to compress images through plane fitting. In this scheme, the image is firstly divided into non-overlapping blocks (typically of 8×8 pixels). Then, each block is represented by three coefficients. In the decoding step, the retrieved image can be obtained through these coefficients. Hence, their scheme can achieve high compression ratio.

This paper provides an effective fractal image compression algorithm, which utilizes the plane fitting to decide whether a domain block is similar enough to a given range block. First, three coefficients are computed for describing the range and domain blocks. Then, the domain blocks matching best with every range block are derived by analyzing the relation between their coefficients. Experimental results show that the proposed method can shorten encoding time markedly, while the retrieved image quality is still acceptable. In the decoding step, a kind of simple line fitting on block boundaries is used to reduce blocking effects. At the same time, the proposed method can also achieve high compression ratio.

This paper is organized as follows. In Section 2, the conventional fractal image compression method is described. Section 3 discusses briefly the algorithm of plane fitting proposed by Ameer *et al.*[15] Section 4 presents a novel fractal image compression method. Some experimental results to verify the performance of the proposed method are given in Section 5. Finally, conclusion is made in Section 6.

2. Fractal image encoding

The conventional fractal image encoding is based on contractive transformations and a partitioned iterated function system. First, the original image I with the size of $M \times N$ is partitioned into non-overlapping $R \times R$ blocks called range blocks and overlapping $D \times D$ blocks called domain blocks. The size of domain blocks must be greater than the size of range

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61173183, 60973152, and 60573172), the Special Scientific Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20070141014), and the Natural Science Foundation of Liaoning Province, China (Grant No. 20082165).

†Corresponding author. E-mail: wangxy@dlut.edu.cn

© 2012 Chinese Physical Society and IOP Publishing Ltd

blocks to ensure that the transformations are contractive. Usually, we set $D = 2R$. All the range and domain blocks can be obtained using a sliding window method. Figure 1 shows the arrangement of range and domain blocks. Then, each domain block is associated with 8 isometries including identity, 90° clockwise rotation, 180° clockwise rotation, 270° clockwise rotation, x reflection, y reflection, $y = x$ reflection, and $y = -x$ reflection. The domain block pool is denoted by Ω. Finally, for each range block R_i $(1 \le i \le (M \times N)/(R \times R))$, we search the domain block pool Ω to obtain a domain block D_i matching with R_i best and a contractive mapping w_i which must ensure that w_i (D_i) has the minimum difference from R_i. As a result, we obtain a set of contractive affine transformations: $w = \{w_i,\ 1 \le i \le (M \times N)/(R \times R)\}$. If all w_i are contractive, w is also contractive. Based on the contractive mapping fixed-point theorem, there would be one unique image I^* and $w(I^*) = I^*$. The collage theorem guarantees that I^* will approximate to the original image I. In this way, the set w constitutes a PIFS and its parameters are the fractal codes for I.

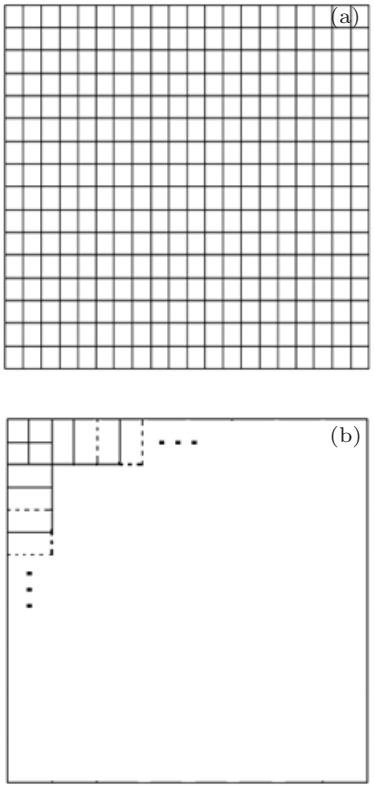


Fig. 1. Arrangements of (a) range and (b) domain blocks.

The most time-consuming step of the encoding process is to search for a domain block in Ω, because every range block must be compared to all domain blocks to find the most suitable one. However, the calculation of similarity between a range block and a domain block is complex.

The reconstruction process is a relatively simple iterative process. By the nature of fractal images, any image can be used as the initial image, and the original image can be generated after many decompression iterations.

3. Plane fitting

In Ameer *et al.*'s paper,[15] the image is divided into non-overlapping blocks (typically of 8×8 pixels), with each considered as a three-dimensional (3D) surface. The z axis is the pixel gray value. The simplest surface is a plane, i.e., $z = ax + by + c$. The block centre is selected as the origin. The goal is to find the coefficients a, b, and c, which best fit the given block. The minimum matching error function becomes

$$\min_{a,b,c} \sum_{x=1}^{N} \sum_{y=1}^{N} \{af(x) + bf(y) + c - g(x,y)\}^2, \quad (1)$$

where N is an even number representing the block dimension, $g(x,y)$ is the pixel value, and

$$f(t) = \begin{cases} t - \dfrac{N}{2}, & \dfrac{N}{2} < t \le N, \\ t - \dfrac{N}{2} - 1, & 1 \le t \le \dfrac{N}{2}. \end{cases} \quad (2)$$

Setting the derivatives of Eq. (1) with respect to a, b, and c to be zero, we have

$$a = \frac{\sum\limits_{x=1}^{N} \sum\limits_{y=1}^{N} f(x)g(x,y)}{N \sum\limits_{x=1}^{N} f(x)^2}, \quad (3)$$

$$b = \frac{\sum\limits_{x=1}^{N} \sum\limits_{y=1}^{N} f(y)g(x,y)}{N \sum\limits_{y=1}^{N} f(y)^2}, \quad (4)$$

$$c = \frac{\sum\limits_{x=1}^{N} \sum\limits_{y=1}^{N} g(x,y)}{N^2}. \quad (5)$$

From Eq. (5), we find that c is the average pixel value of the block.

4. Proposed fractal image compression technique

Figure 2 shows the test image Zelda (512×512). First, we divide image Zelda into non-overlapping

range blocks (8×8), and overlapping domain blocks (16×16). The step length of domain blocks is 8 pixels, so the number of domain blocks is $(512/8-1)^2 = 4225$. Then, a, b, and c of every range and domain block are computed using Eqs. (3), (4), and (5).

Figure 3 shows the distributions of a and b in the range and domain blocks of Zelda. It can be seen that the distributions of a and b in the range and domain blocks are very similar. They are all approximately symmetrical and the absolute values of a and b of most blocks are between 0 and 5. Figure 4 illustrates the eight transformations (t_j, $j = 0, 1, \ldots, 7$) used in the proposed method. A block z_i is expressed by three parameters (a_i, b_i, c_i). For example, $t_2(z_i) = z_i^* = (-a_i, -b_i, c_i)$. The c_i is the average pixel value of the block. The eight transformations do not change the c_i of block.



Fig. 2. Original image of Zelda.

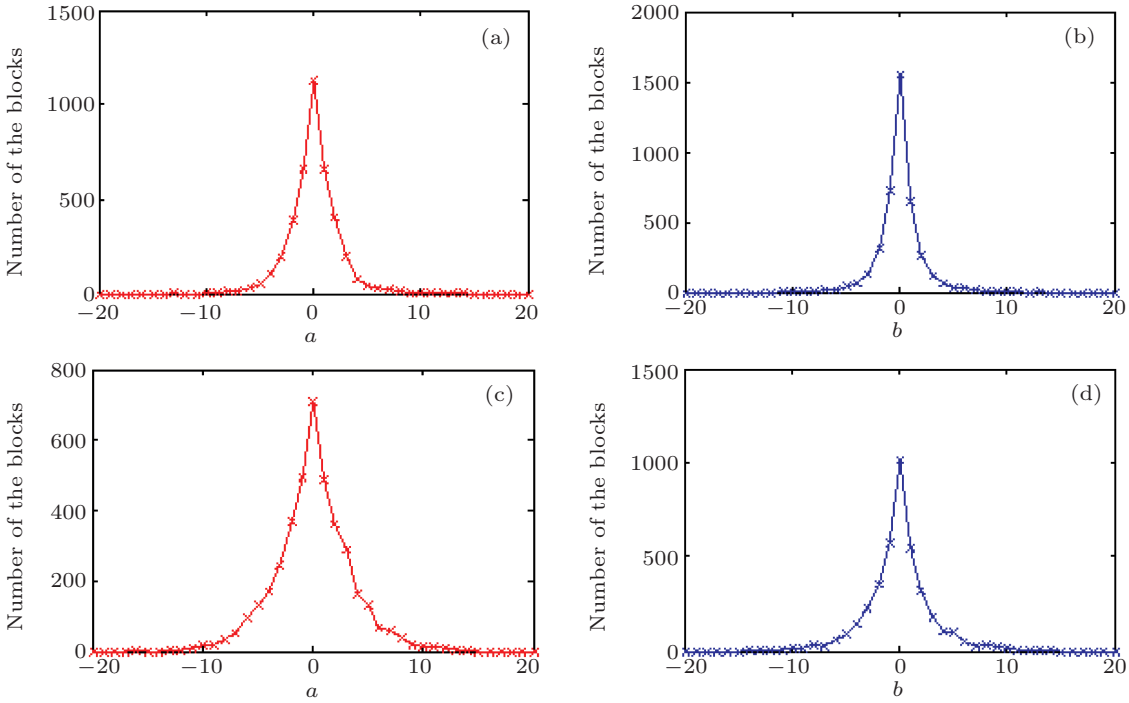


Fig. 3. (colour online) (a) Distribution of a in range blocks; (b) distribution of b in range blocks; (c) distribution of a in domain blocks; (d) distribution of b in domain blocks. For statistical purpose, both a and b are rounded off. For example, in Fig. 3(a), the number of blocks is 1130 when a is about 0 rather than exactly 0.

In the proposed method, the similarity degree of a range block R_i and a domain block D_k among Ω is defined as

$$E_{i,k} = \min_j \{ |a(R_i) - t_j(D_k)\, a| + |b(R_i) - t_j(D_k)\, b| \}, \quad (6)$$

where $j = 0, 1, \ldots, 7$, and $a(R_i)$ and $b(R_i)$ represent the a and b of the block R_i. For every range block R_i, the domain block with the highest similarity degree is the one that results in the lowest error E. Moreover, the upper limit of E is denoted as E_{ceiling}. A Boolean variable named "Find" is also used. If the lowest E for R_i is still larger than E_{ceiling}, the "Find" for this range block will be recorded as false, or else it will be true. When the best one D_{B} (the domain block with the highest similarity degree) is found and the "Find" is true, some information should be recorded. They include (i) the positions of D_{B} (X and Y); (ii) the

transformation of D_{B} (j); (iii) the shift of c (Δc), i.e., the difference between D_{B}'s c and R_i's c.

The parameters X, Y, j, and Δc form the codes for R_i. If the Boolean variable "Find" for the range block is false, then no information is recorded but "Find" itself. When the codes of all range blocks are recorded, the encoding process is over. The proposed encoding process can be divided into five steps as follows:

i) load the original image;

ii) divide I into non-overlapping range blocks and overlapping domain blocks;

iii) compute a, b, and c of every range block and domain block;

iv) for each range block, find D_{B} among the domain pool;

v) record X, Y, j, and Δc as the fractal codes of every range block.

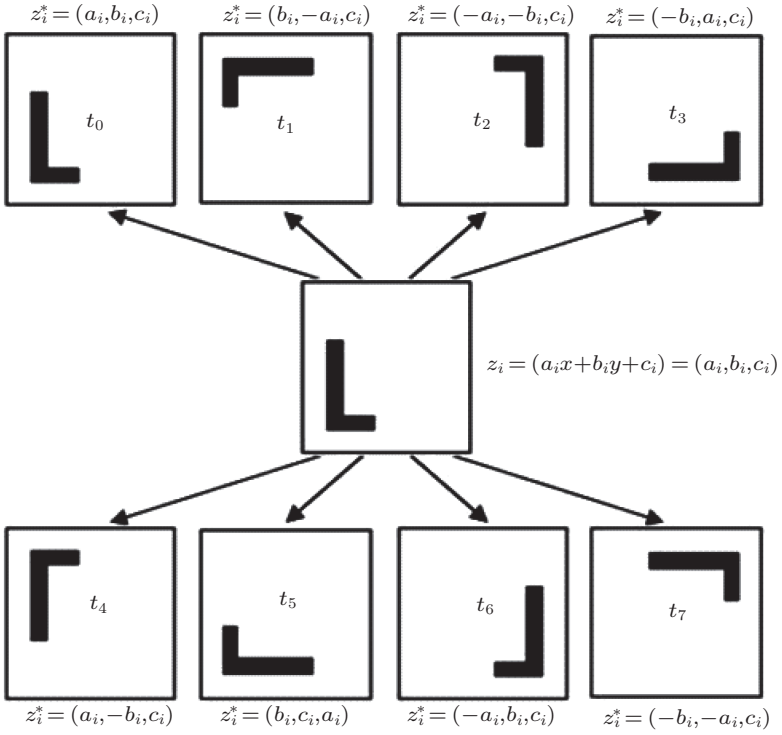


Fig. 4. Eight transformations of the proposed method.

The decoding process is a simple iterative process. Any image can be the initial image. To reconstruct a range block, the fractal codes for that block are read firstly. The X, Y, j, and Δc are then separated. The corresponding domain block is compressed to the size of the range block. Afterwards, the j and Δc are used on the domain block. The result is stored as the range block. When all of the range blocks are reconstructed, one iteration is finished and the previous range image acts as a domain image for the next iteration. In general, the resulting image resembles the original image when the number of iterations is 8. However, when the "Find" for the range block is false, the decoding process is different. Figure 5 shows the solution.

In Fig. 5(a), the E_{ceiling} is 0.3. There are 37 range blocks whose "Find" is false, while the total number of the range blocks is 4096. In Fig. 5(b), the area "S" is supposed to be one of the blocks in Fig. 5(a), and the area "Up" (or "Down", "Left", "Right") denotes a row (or a column) of the neighboring blocks. In an iteration of the decoding process, the range blocks whose "Find" are true are computed firstly. Then, other blocks like "S" are computed through the following steps:

(1) $S(x,y)=0$, $x,y=1,2,\ldots,8$;

(2) $S(x,*)=\mathrm{Up}/2$, $x=1,2,\ldots,8$, $y=1,2,3,4$;

(3) $S(x,*)=\mathrm{Down}/2$, $x=1,2,\ldots,8$, $y=5,6,7,8$;

(4) $S(*,y)=\mathrm{Left}/2$, $x=1,2,3,4$, $y=1,2,\ldots,8$;

(5) $S(*,y)=\mathrm{Right}/2$, $x=5,6,7,8$, $y=1,2,\ldots,8$,

where $S(1,1)$ means the pixel value of the top left corner of "S", $S(8,8)$ means the pixel value of the bottom right corner of "S", $S(1,0)$ means the pixel value of the top corner of "Left", $S(0,1)$ means the pixel value of the left corner of "Up", and so on.

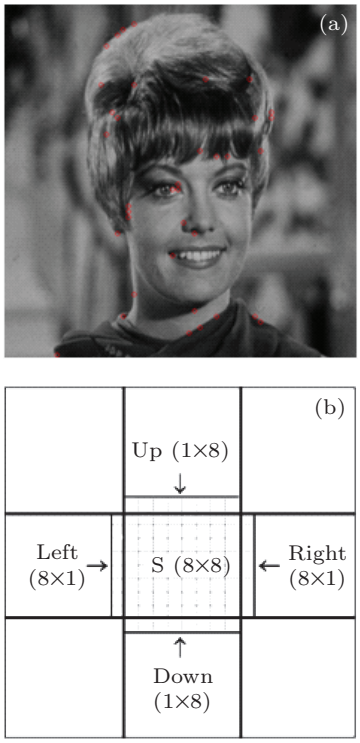


Fig. 5. (a) The range blocks whose "Find" is false. These areas are marked by circles. (b) Diagram of the solution. The area "S" denotes one of the blocks in (a).

5. Experimental results

Six 512×512 test images are shown in Fig. 6. Several experiments were conducted to analyze the performance of the proposed algorithm. The quality of the retrieved image g in comparison to the original image f is measured by the peak signal to noise ratio (PSNR), which is given by

$$Q_{\mathrm{PSNR}} = 10 \times \lg \left(\frac{255^2}{\left(\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(f\left(i,j\right) - g\left(i,j\right) \right)^2 \right)} \right), \quad (7)$$

where N is the image size. Figure 7 illustrates the PSNR versus the number of iterations. From Fig. 7, it is seen that when the number of iterations is about 10, the PSNRs of all the test images have been stabilized. More detailed results are given in Table 1. All the experiments were implemented on a personal computer using Visual Basic 6.0 language.

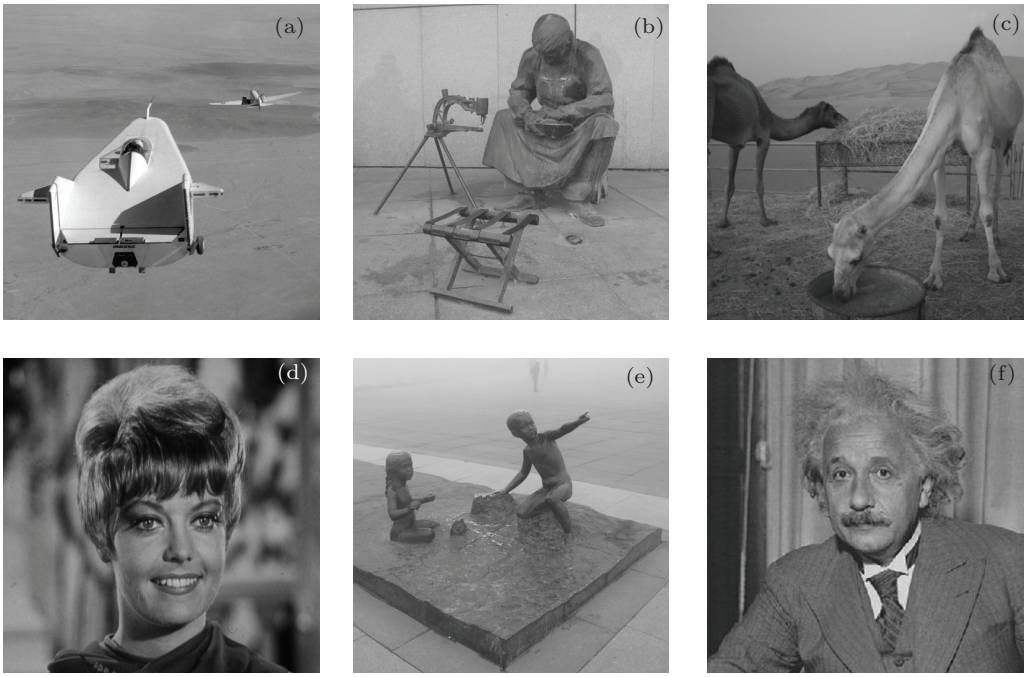


Fig. 6. Various original test images. (a) Lifting body, (b) cobbler, (c) dromedary, (d) Zelda, (e) kids, and (f) Einstein.

To reduce blocking effects, the block boundaries are modified after the decoding process. In Ref. [15], a first-order curve fitting was used on four pixels (two from each block) to estimate the boundary pixels better. Mathematically,

$$\hat{g}\left(Nx, y\right) = r - m, \quad \hat{g}\left(Nx+1, y\right) = r + m, \quad (8)$$

where $m = \boldsymbol{v}\boldsymbol{k}^{\mathrm{T}}/\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$, $\boldsymbol{v} = [-4, -1, 1, 4]$, $\boldsymbol{k} = [\hat{g}(Nx-1, y), \hat{g}(Nx, y), \hat{g}(Nx+1, y), \hat{g}(Nx+2, y)]$, r is the average of k, $\hat{g}$ is the retrieved image, $x = 1, 2, \ldots, X/N$, $y = 1, 2, \ldots, Y$, X and Y are the image

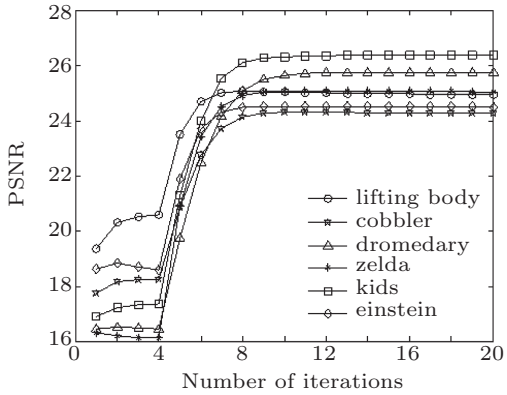


Fig. 7. PSNR versus the number of iterations. The size of the range blocks is 8×8. The E_{ceiling} is 0.3.

size, and N is the size of the range blocks. Figure 8 presents the decoding results after reducing blocking effects.

Table 1. Performance of the proposed method on the set of test images. The size of the range blocks is 8×8 and the E_{ceiling} is 0.3.

Image	PSNR/dB	Compression ratio	Coding time/s
lifting Body	25.05584	21.1689	4.1718
cobbler	24.29265	20.7914	3.8906
dromedary	25.72531	20.7227	3.8437
Zelda	25.07729	20.6585	3.7031
kids	26.36989	20.7610	3.9375
Einstein	24.51353	20.8646	3.7812

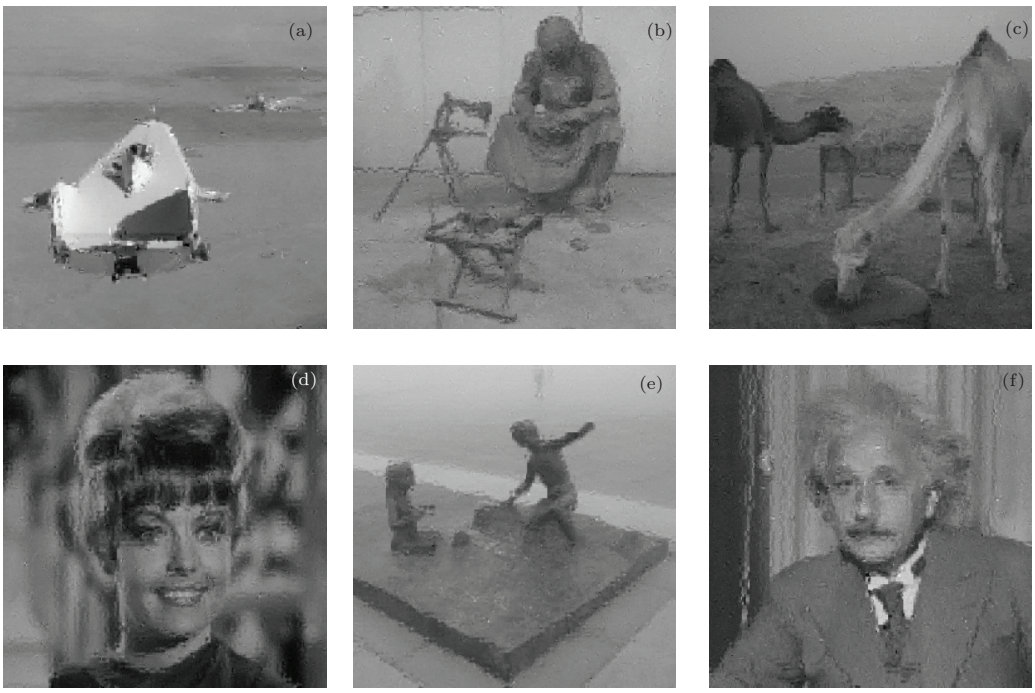


Fig. 8. Decoding results after reducing blocking effects for (a) PSNR=25.53009, (b) 24.86744, (c) 26.36989, (d) 25.98236, (e) 26.82747, and (f) 25.29779.

The distribution of Δc in the encoding process of image Zelda is shown in Fig. 9. In order to improve the compression ratio of the proposed method, Huffman coding can be used to compress Δc before the encoding process. The compression ratios of test images when Huffman coding is used are given in Table 2.

The Huffman table used in Table 2 was constructed from measurements on several images. In addition, the value of E_{ceiling} can also affect the PSNR of the retrieved image. Take the image Zelda for example, the relationship between PSNR of retrieved image of Zelda and the value of E_{ceiling} is given in Fig. 10. From Fig. 10, it can be seen that, the PSNR reaches its peak value when the E_{ceiling} is 0.3.

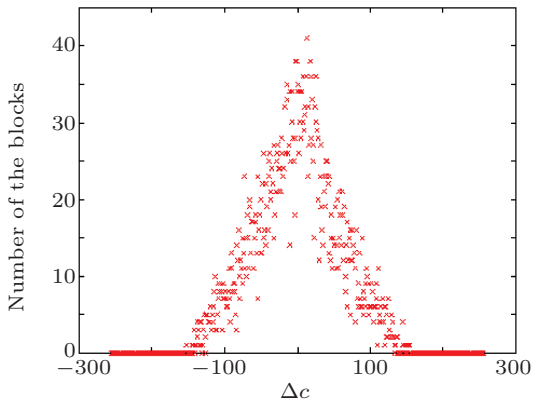


Fig. 9. (colour online) Distribution of 25.29779 in the encoding process of image Zelda.

Table 2. Compression ratios when Huffman coding is used on Δc.

	lifting body	cobbler	dromedary	Zelda	kids	Einstein
Without Huffman coding	21.1689	20.7914	20.7227	20.6585	20.7610	20.8646
With Huffman coding	22.8154	23.0213	22.8127	22.6237	22.8430	22.7414

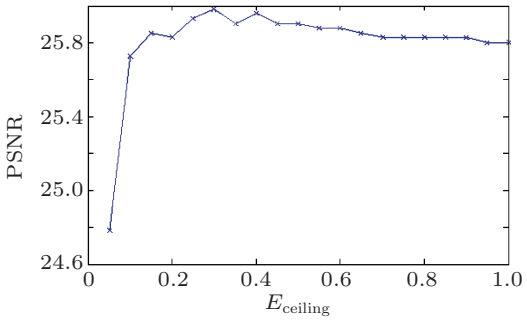


Fig. 10. (colour online) PSNR of the retrieved image Zelda versus the value of E_{ceiling}.

6. Conclusion

In this paper, we have examined the basic fractal image compression method and introduced the theory of plane fitting, then a new fractal image coding method based on plane fitting is proposed. In the proposed algorithm, a simple but very efficient similarity measure method is given. The results show that this method can speed up the encoding process greatly and obtain a high compression ratio, while the retrieved image quality is still relatively acceptable.

References

[1] Barnsley M F 1988 *Fractal Everywhere* (New York: Academic) p. 34
[2] Jacquin A E 1992 *IEEE Trans. Image Process.* **1** 18
[3] Fisher Y 1995 *Fractal Image Compression: Theory and Application* (New York: Springer-Verlag) p. 121
[4] Duan L M, Shao C F, Yue H and Guan H Z 2010 *Acta Phys. Sin.* **59** 4499 (in Chinese)
[5] Sze C J, Liao H Y, Fan K C, Chern M Y and Tsao C K 1996 *Image Vision Comput.* **14** 401
[6] Zhang X, Ren W, Tang D N and Tang G N 2010 *Acta Phys. Sin.* **59** 5281 (in Chinese)
[7] Truong T K, Jeng J H, Reed I S, Lee P C and Li A Q 2000 *IEEE Trans. Image Process.* **9** 529
[8] Yue H, Shao C F and Yao Z S 2009 *Acta Phys. Sin.* **58** 4523 (in Chinese)
[9] Lai C M, Lam K M and Siu W C 2003 *IEEE Trans. Image Process.* **12** 1398
[10] Song Y R and Jiang G P 2009 *Acta Phys. Sin.* **58** 5911 (in Chinese)
[11] Duh D J, Jeng J H and Chen S Y 2005 *Image Vision Comput.* **23** 1115
[12] Distasi R, Nappi M and Riccio D 2006 *IEEE Trans. Image Process.* 15 89
[13] Li X G, Gao Z Y, Zhao X M and Jia B 2008 *Acta Phys. Sin.* **57** 4777 (in Chinese)
[14] Wang X Y and Wang S G 2008 *Comput. Graph.* **32** 445
[15] Ameer S and Basir O 2009 *Image Vision Comput.* **27** 385